RESEARCH ARTICLE

Journal of Problem Solving

Solving Large Problems with a Small Working Memory

Zygmunt Pizlo[1] and Emil Stefanov[1]

[1] Department of Psychological Sciences, Purdue University

Correspondence:
Zygmunt Pizlo, Department of Psychological Sciences, Purdue University, West Lafayette, IN 47907-2081; Phone: (765) 494-6930.
Email: pizlo@psych.purdue.edu

Keywords:
Traveling Salesman Problem, visual pyramid model, working memory, computational model

We describe an important elaboration of our multiscale/multiresolution model for solving the Traveling Salesman Problem (TSP). Our previous model emulated the non-uniform distribution of receptors on the human retina and the shifts of visual attention. This model produced near-optimal solutions of TSP in linear time by performing hierarchical clustering followed by a sequence of coarse-to-fine approximations of the tour. Linear time complexity was related to the minimal amount of search performed by the model, which posed minimal requirements on the size of the working memory. The new model implements the small working memory requirement. The model only stores information about as few as 2–5 clusters at any one time in the solution process. This requirement matches the known capacity of human working memory. We conclude by speculating that this model provides a possible explanation of how the human mind can effectively deal with very large search spaces.

INTRODUCTION

The number of situations (states) in the game of chess is much larger than the number of neurons in the brain, and even larger than the number of atoms in the universe. This is also true with many interesting and important tasks, such as solving mathematics and physics problems, or scheduling a trip for visiting a number of places. These problems are called *computationally intractable* because producing an optimal solution may require exploring the entire search space and the time needed to do this is impractically long. To get an intuition about the nature and magnitude of the search involved, assume that one is able to examine a billion (10^9) states per second and one started at the big bang, which occurred 13.7 billion years ago. By continuing this search till the present day, one would be able to examine as many as 4×10^{26} states. As you will see, however, the combinatorial optimization problems, which untrained undergraduate or even elementary school students can solve nearly-optimally within a minute or two, contain that many or even more states. Considering the fact that human working memory can store and manipulate only a few items at a time, and that a human being cannot make more than a few comparisons per second, the question is not how humans can explore substantial portions of such large search spaces, but how they can do anything meaningful with such spaces in the first place, not to mention produce near-optimal solutions? Understanding mental mechanisms that allow doing just that has been the ultimate goal of cognitive science ever since its inception some 70 years ago (Gardner, 1987). In this paper, we provide a possible explanation of these mechanisms and present a computational model whose performance matches that of human subjects. The model presented here is an elaboration of our previous models. The main new feature of the proposed model is the small amount of memory used.

We focus our study on one particular hard classic problem: the Traveling Salesman Problem (TSP). The TSP is one of the most well known challenging NP-complete problems in computational complexity theory (Lawler, Lenstra, Rinnooy Kan, & Shmoys, 1985). Additionally, there is a reasonably large body of empirical evidence about how well humans produce TSP tours (see MacGregor & Chu, 2011, for a review).

OVERVIEW OF THE TSP PROBLEM

The TSP problem consists of N points (called cities) on a plane. The goal is to produce a closed path (called a tour) of minimal length that goes through each city exactly once. An example is shown in Figure 1. The problem is equivalent to determining a closed polygon connecting all points, whose perimeter is shortest.

The TSP problem frequently occurs in nature when animals, including humans, perform simple visual navigation tasks. Let us take a very simple example: If one begins in Paris, and sets out to visit several places in Germany, Hungary and Italy before going back to Paris, it is obvious that in order to minimize the total travel length, all places in Germany should be visited one after another before going to Hungary and finally to Italy. The decision about the order in which the three countries are visited is made first and it should be the one just mentioned, or its reverse, namely, Italy, Hungary and Germany (Figure 2). It is fairly obvious that going from Paris to Hungary, Germany, Italy, and back to Paris is sub-

http://dx.doi.org/10.7771/1932-6246.1155

optimal, regardless of the order in which the places within each country are visited. The quality of the global aspects of the tour are unaffected by its local aspects. If the global tour is optimal, local errors will not be able to make the overall tour very much longer than the optimal one. And conversely, if there is an error in the global tour, the overall tour length will be highly suboptimal, regardless of the optimality of its local parts.

Humans solve TSP problems not only when we plan our vacation. We solve TSP (or closely related problems) when we walk around a town, buy food in a grocery store, or walk between and around buildings. Other animals also solve TSP when they forage—this includes lower animals such as bumblebees and hummingbirds, which are able to optimally plan a tour of places containing food (Lihoreau, Chittka, & Raine, 2010, 2012).

DIFFICULTY OF THE TSP PROBLEM

For an N-city problem there are $T = (N - 1)! / 2$ possible tours. The number of possible tours T grows very rapidly with the number of points N. For example, for $N = 6$, $T = 60$ and for $N = 16$, $T = 6\times10^{11}$. Note that for 16 points, the number of tours is already larger than the number of neurons in the human brain (Azevedo et al., 2009). It follows that even for moderate values of N, the memory of a person solving the TSP will not be large enough to store substantial portion of the problem space. Surprisingly, humans need only a few minutes to produce near-optimal tours for $N = 100$ (Pizlo, Stefanov, Sallweachter, Li, Haxhimusa, & Kropatsch, 2006; Dry, Lee, Vickers, & Hughes, 2006), despite the fact that the number of possible tours is truly astronomical (about 10^{155}). The way the human brain works when it solves difficult prob-

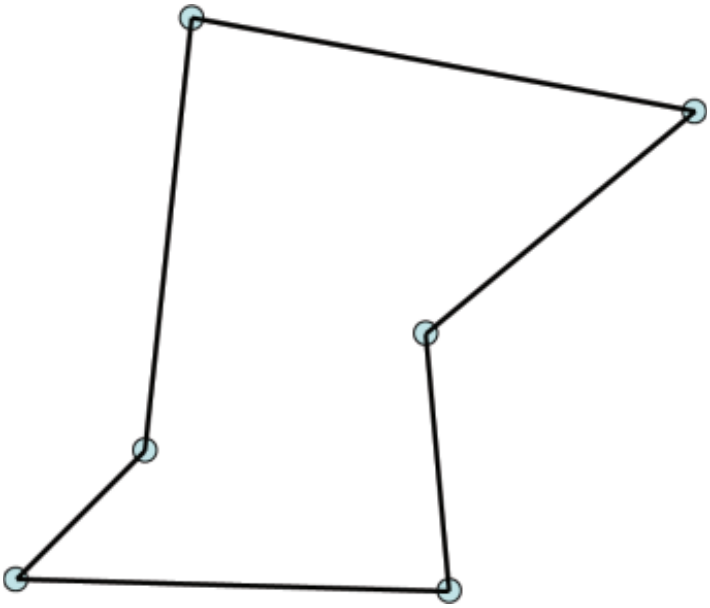

Figure 1.
The Traveling Salesman Problem refers to the task of finding the shortest tour of a number of cities (here 6 cities marked by points). The number of possible tours for even a moderately large number of cities is astronomically high.

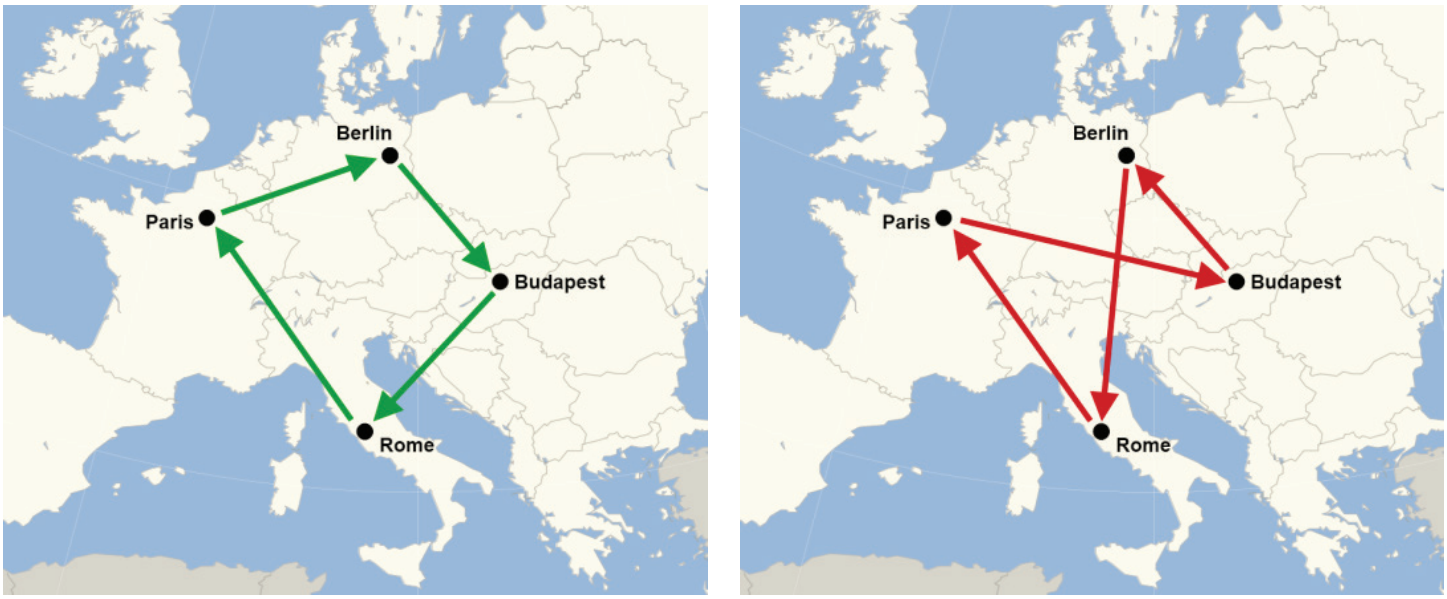

Figure 2.
Left: the optimal order of visiting France, Germany, Hungary and Italy. Right: a non-optimal order of visiting these four countries. If the task is to tour several cities in each of these four countries, then visiting cities in one country before going to the next country is likely to lead to an optimal tour, as long as the order in which the four countries are visited is like that on the left. If these four countries are visited in the order shown on right, the tour is likely to be highly suboptimal regardless of which cities are visited in each of these countries. This is related to the fact that once an error is made in the global aspects of the tour, this error cannot be undone on a more local level.

lems continues to surprise and impress researchers in both Cognitive Psychology and Artificial Intelligence. The underlying mechanisms remain a mystery. Somehow, instead of performing a substantial amount of search in the problem space, the brain changes the representation of the problem so that a minimal amount of search in the new representation can produce a near-optimal solution to the original problem. The fact that changing the problem representation is a critical step in solving difficult problems was pointed out about 100 years ago by the gestalt psychologists (Duncker, 1945; Wertheimer, 1945). However, no formal theory of how the human mind does it has ever been proposed. In this paper we explain how a near-optimal solution to combinatorial optimization problems, such as the TSP, can be produced quickly by a brain whose size is many orders of magnitude smaller than the size of the problem and using working memory that can store and manipulate only a few items at a time.

HUMAN PERFORMANCE ON A EUCLIDEAN TSP

Figure 3 shows how a randomly generated 10-city TSP was solved by four subjects. An optimal tour is also shown. The percentages inside the figures represent how much longer the tour produced by a subject was compared to the shortest tour.

All five tours are different, but differences are not very large. For example, the cluster of three cities on the right was treated as a cluster by all four subjects. Namely, these three cities were visited one after another, although not necessarily in the same order. These three cities were also visited one after another in the optimal tour. It took three of the four subjects less than 15 seconds to produce the tour. OSK used more time: on average, he produced a tour in a 10-city TSP in 50 seconds. OSK solved 85% of 10-city problems optimally (there were 25 randomly generated problems). The other three subjects produced 65–75% optimal tours in 10-city problems. On average, the error produced by the subjects was between 0.3% and 0.9%. In other words, if the length of the optimal tour is 100m, our subjects produced a tour whose length, on average, was less than 101m. The average error was computed using all tours, including the optimal tours. Considering the fact that a large proportion of tours were optimal, these tours contributed 0% to the average error.

Figure 4 shows tours produced by the same four subjects in a 20-city TSP. ZP's tour differed from the optimal tour in the placement of only one city, which resulted in 0.4% error. This time, OSK's tour was about 11% longer than the optimal tour. It took each subject twice as much time to solve a 20-city TSP compared to a 10-city TSP. This indicates that the complexity of the mental mechanisms is, on average, linear. OSK produced 40% optimal tours. The other three subjects produced 10–25% optimal tours. The average error of the 4

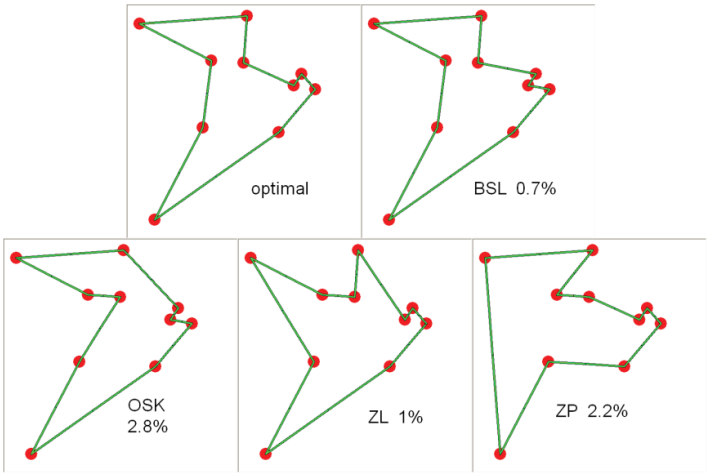

Figure 3.
Tours produced by four subjects in a randomly generated 10-city TSP problem (results from experiments reported by Pizlo et al., 2006).

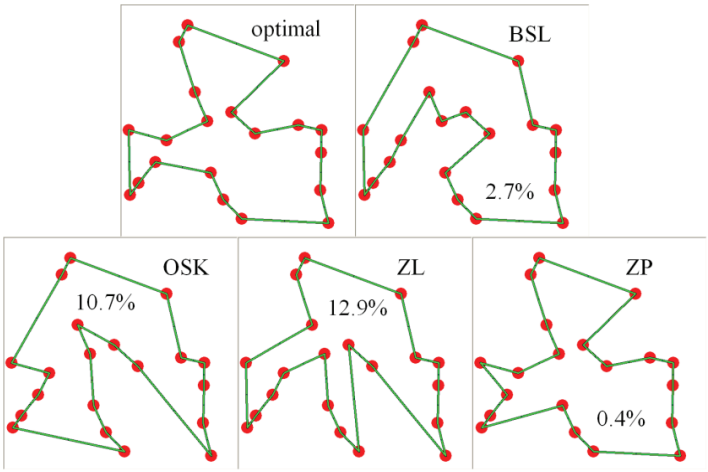

Figure 4.
Tours produced by four subjects in a randomly generated 20-city TSP problem.

subjects ranged between 1.5% and 4% (statistical analysis of these results can be found in Pizlo et al., 2006).

Finally, Figure 5 shows how a randomly generated 50-city TSP was solved by four subjects. Note that OSK did produce an optimal tour. Again, the percentages inside the figures represent how much longer the tour produced by a subject was compared to the shortest tour. It took each subject 5 times as much time to solve a 50 city TSP compared to a 10-city TSP. This fact confirms our earlier observation about the linear complexity of the mental mechanism and it is consistent with a report by Dry et al. (2006). For a 50-city problem, it is quite rare for a subject to produce an optimal tour. The average error ranged between 2.5 and 5%.

Now that we (and others) had collected results characterizing human performance on the TSP, the next step was to formulate a computational model of the underlying mental mechanisms. There are two types of cognitive abilities that

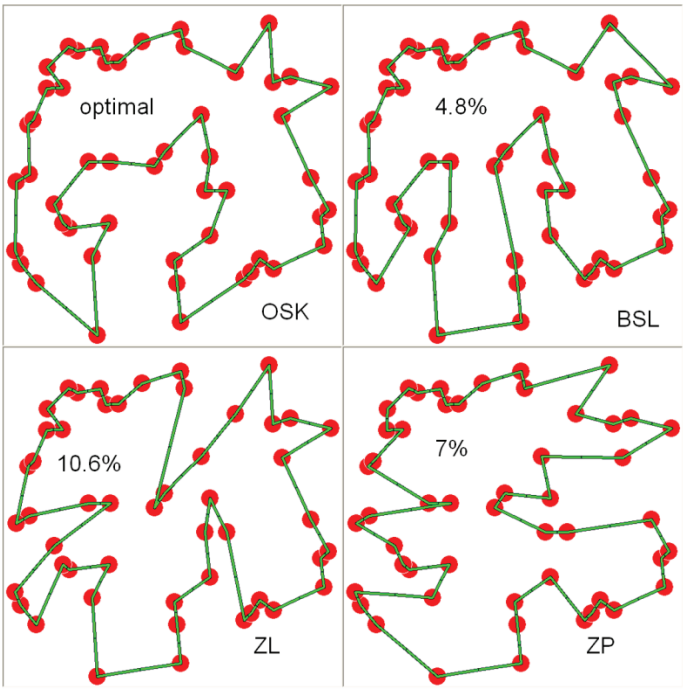

Figure 5.
Tours produced by 4 subjects in a randomly generated 50 city TSP problem.

lead to two quite different modeling strategies. The first type corresponds to problems that are computationally easy. For example, estimating the intensity of light falling on the retina or remembering several items in one's working memory are computationally easy. It is not difficult to come up with several models of each of these abilities, but it is difficult to decide which, if any, of the proposed models faithfully represents the cognitive mechanism under study. The decision is not easy because studying mental events is a *black-box problem*. The researcher measures input-output relations in order to infer the architecture and the algorithms inside the box. No matter how hard he tries, there will always be unresolved uncertainty and ambiguity. So, the researcher must rely on one or another type of model selection methods (Pitt et al., 2003). The second type of cognitive abilities corresponds to problems that are computationally difficult. For example, detecting contours of 3D objects in 2D retinal images, recovering 3D shapes from 2D images, or motor control. The human mind performs these functions very well, and yet explaining how the mind does it has proven nearly impossible. In fact, these were precisely the types of difficult and amazing cognitive functions that inspired the imagination of engineering, neuroscience and psychological communities in the early 1950s, giving rise to what we now refer to as the Cognitive Revolution. These cognitive functions are computationally so difficult that it may very well be that there is only one way to solve them. This offers a unique opportunity to a researcher, because once he manages to formulate a computational method which emulates human performance in a given cognitive task, he may have good reason for optimism that his model not only emulates performance, but also emulates the underlying cognitive mechanisms. This will reduce the effort and uncertainty related to model selection. No doubt, it will always be beneficial to obtain supporting psychological and/or neurophysiological evidence, as well.

It was quite obvious to us and to others that solving TSP problems belongs to the second type of cognitive functions. How does one emulate linear-time complexity and, at the same time, guarantee near optimal performance? Subjective reports of our subjects about attempts to find and use clusters made sense on rational grounds. Interestingly enough, these reports were consistent with what we already knew about the anatomy of the human visual system. The way we (and others) presented TSP to our subjects made this a visual problem and it was reasonable to assume that subjects treated the task visually. This speculation received support from surprisingly low individual variability and small, if any, effect of practice and learning (van Rooij et al., 2006). These characteristic are quite distinctive for visual, as opposed to higher level cognitive mechanisms. Next, we will describe some known facts about the anatomy and physiology of the human visual system and discuss them in terms of a mutli-resolution/multiscale pyramid algorithm that provided the basis for our previous and our new model of how humans solve TSP problems.

HUMAN VISUAL SYSTEM AS A MULTISCALE PYRAMID

Figure 6 illustrates the architecture of a pyramid model (see Jolion & Rosenfeld, 1994, for an excellent review of these models). The bottom layer represents the retinal or the cam-

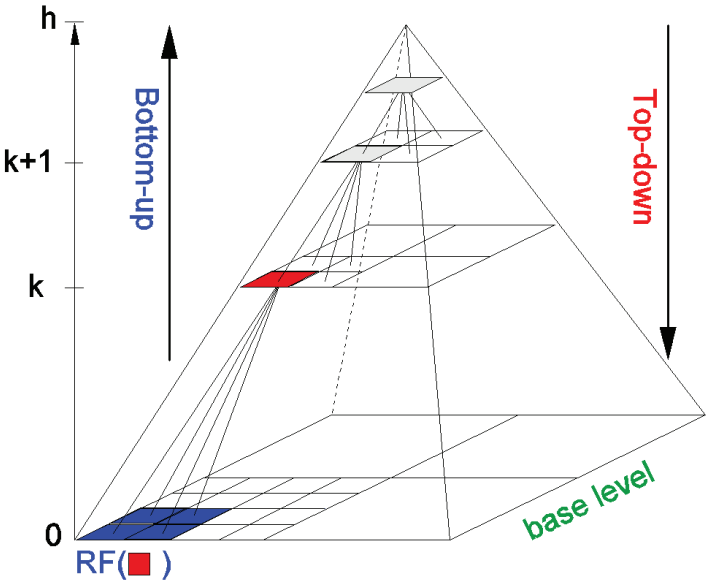

Figure 6.
Schematic illustration of a visual pyramid.

era image. In the human retina there are 6 million cones that provide input for most visual functions such as pattern and object perception, reading, color, motion, and depth. The second layer in the pyramid contains parent cells (nodes), each parent receiving input from four child cells. The third layer is the grandparent layer and the spatial relation between grandparent and parent cells is the same as between parent and children cells. This repeats until there is only one cell at the top of the pyramid. The cells in the pyramid function simultaneously and independently. This makes pyramid architecture a version of a parallel computer.

The pyramid can perform operations in both bottom-up and top-down directions. Both types of operations are important. Bottom-up operations provide a way to estimate statistical information from the image, such as intensity, color, texture, and motion. Top-down operations are responsible for estimating spatially global characteristics such as the shapes and sizes of objects in the image. It is typically assumed that each cell in the pyramid can only perform some simple operations and has limited memory. As a result, if there is a need to extract information about a spatial detail, like the position of a point relative to another point, the pyramid will have to proceed in a top-down (coarse-to-fine) direction so that the cells that can "see" large parts of the image provide a global coordinate system for the cells on the lower layers of the pyramid, making it possible to perform precise estimates of large spatial extents (Pizlo, Rosenfeld, & Epelboim, 1995). Essentially, larger clusters that are established closer to the top of the pyramid guide the decisions and computations within smaller clusters, closer to the bottom of the pyramid. If the main operation performed by the cells of the pyramid is the computation of the mean value of the intensity, the resulting pyramid is called multiscale/multiresolution (Burt & Adelson, 1984). Such pyramids are also known under the name of wavelet pyramids, quite popular in image and video compression applications. It is now commonly accepted that the human visual system is a pyramid, whose cells can perform multiple operations, not only the computation of the first and second statistical moments (Wandell, 1995). Neurons in progressively higher stages of the visual system have larger receptive fields, which means that they receive information from larger parts of the retina. Besides growing amount of anatomical and neurophysiological evidence, there is also ample psychophysical evidence indicating that visual algorithms perform operations in a way compatible with pyramid architecture (Pizlo et al., 1995, 1997).

There is one important characteristic of the visual pyramid that is often ignored in computational models. Look at Figure 7. Because of anatomical constraints, the high resolution representation is only available in the center of the retinal image. More precisely, it is known that the distance between the neighboring cones in the human retina gets progressively larger as the distance from the center of the retina increases. As a result, we can visually resolve spatial details only when we look directly at the details, so that their image is projected to the center of the fovea where the density of cones is highest. The farther a given spatial characteristic is from the center of the visual field, the fewer details we see. This non-uniform distribution of cones is compensated by eye movements which bring the object of interest to the center of the visual field. The combination of the non-uniform distribution of cones with eye movements provides an optimal compromise between the anatomical constraints and computational capabilities. The eye movements allow the visual system to select the optimal sampling precision without overloading the brain with too much information. In fact, having high density of cones on the entire retina would not be anatomically plausible, anyway, because it would require the optic nerve that brings the visual messages from the eyes to the brain to be an order of magnitude thicker than what it actually is.

To summarize this brief overview of the visual pyramid, the pyramid provides a possibility of performing parallel computations and analyzing the image using coarse-to-fine and fine-to-coarse operations. In particular, pyramid is very well suited to perform hierarchical clustering in order to capture spatially global characteristics while preserving the information about spatial details. The visual pyramid cannot analyze the entire image at all spatial resolutions because of anatomical con-

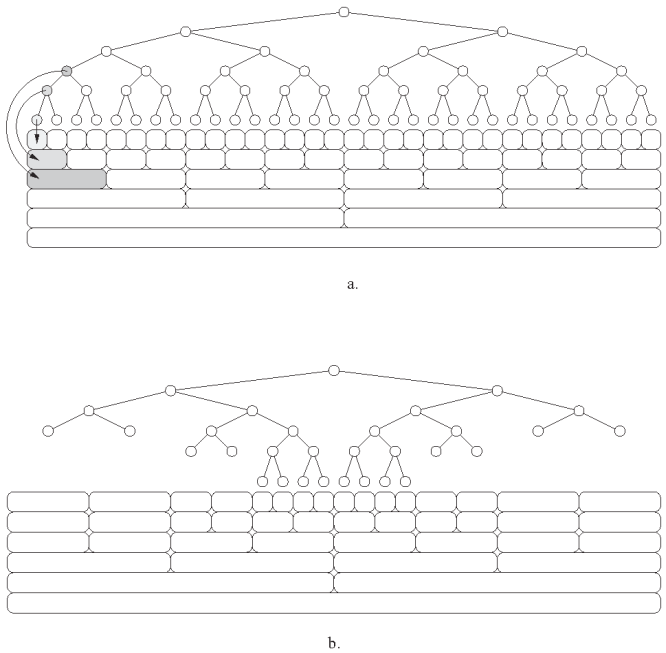

Figure 7.
(a) One-dimensional version of the pyramid shown in Figure 6. (b) Schematic illustration of the pyramid characterizing the human visual system. The high resolution representation on the bottom layers of the pyramid is available only around the center of the retinal image.

straints. In order to integrate information about spatial details across large areas of the image, the observer has to move his attention around the image by using eye movements.

COMPUTATIONAL MODEL

The model described here is an elaboration of our earlier pyramid models (Haxhimusa et al., 2011; Pizlo et al., 2006; Graham et al., 2000). The main new contribution here is that the model uses a working memory that stores only a few pieces of information at a time. Look at Demo 1 (http://www.youtube.com/watch?v=eUBt41mq6PE) for a simple illustration of how our pyramid model solves a TSP using a global-to-local mechanism. The problem with 21 cities has been divided into 6x6 regular arrays. The first step is to determine 2 to 5 clusters using a modified version of the *K*-means algorithm (the number of clusters is a parameter that can be changed). The original *K*-means algorithm determines the optimal partition of a set of points into *K* clusters, where optimality is measured in the least squares sense (Brusco & Stahl, 2005). Figure 8 illustrates the results of applying this algorithm to a simple problem for several values of *K*.

In our new model this algorithm had to be modified because the resolution of the representation (6×6) implies that the individual points may not be visible. In the extreme case, when *N* is large (say larger than 1000), all 36 cells are likely to contain points, so instead of identifying clusters of points, the algorithm identifies high density regions of points. Look at Figure 9, which illustrates the result of our modification of *K*-means algorithm applied to the same example as shown in Figure 8. Figure 10 shows the pseudocode of this algorithm.

In Demo 1, there are three clearly defined clusters. This example was prepared by hand in order to illustrate that our clustering algorithm is able to find clusters that are obvious to a human observer. When the centers of these clusters are connected, the result is the first approximation to a TSP tour. This is not a valid tour because it does not go through all 21 points, but it is useful because it captures the most global aspects of all good tours. In the next step, the model takes the cluster on top right and represents this part of the problem on a 6×6 array. At this stage, the *K*-means algorithm correctly finds five clusters, the individual points. When these points are inserted into the next approximation of the TSP tour, their order in the tour is decided by a "cheapest insertion" criterion, which means that each insertion causes the smallest possible increase in tour length. It is important to point out that minimizing the length of this local part of the tour is not affected by the length of the entire TSP tour. This is of fundamental significance because this means that the measure of distance at these two levels of the problem representation, the coarse level with 3 clusters and the finer level at which the 5 points on the top right are visible, does not have to be the same. This characteristic seems to be compatible with what we know about how humans handle problems with large spaces. A chess player can estimate

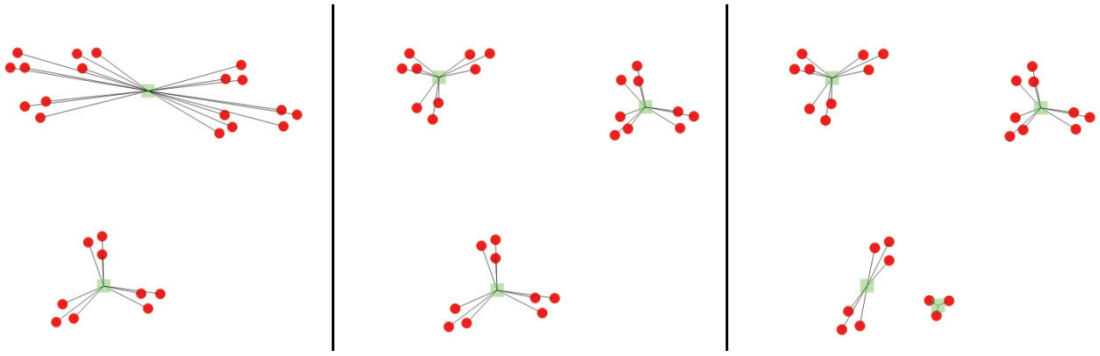

Figure 8.
Regular *K*-means clustering (*K* = 2, 3, 4).

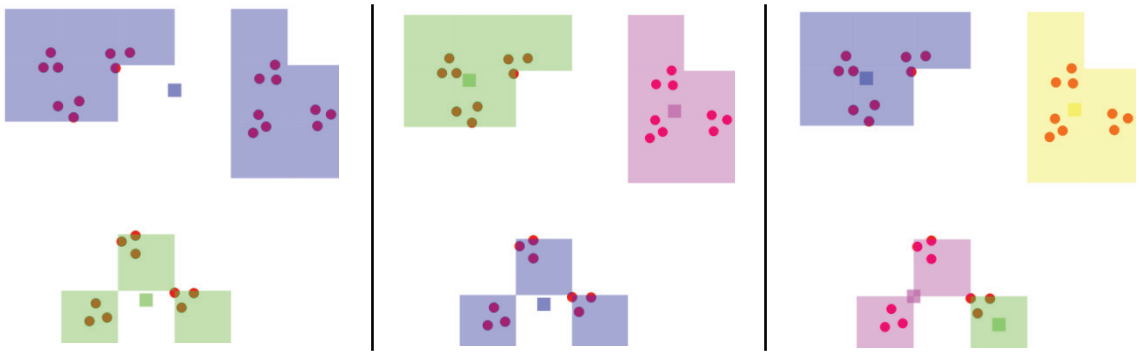

Figure 9.
Image based *K*-means clustering (*K* = 2, 3, 4).

- Each pixel in the image is a point with weight equal to its pixel intensity.
- For $K = 2$ to 5
  - The weighted points are fed into a weighted *K*-means algorithm with the specific *K*.
- Among the four clusterings computed (for $K = 2$ to 5), the one with the largest compactness is selected. Compactness is calculated as:
  - (min inter-cluster distance squared) / (inter cluster distances sum) where:
    - min inter-cluster distance squared = the minimum distance between any two pixels in different clusters
    - inter cluster distances sum = sum of the distances squared between all two points in the same cluster
  - Distances are calculated probabilistically where the probabilities are the pixel intensities. Hence the distance between two pixels at locations (x_1, y_1), (x_2, y_2) with intensities p_1 and p_2 is: $\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)} \times p_1 \times p_2$.
- Note that the centroids outputted are not necessarily in the center of the cluster because the weights might cause them to shift towards certain pixels.

Figure 10.
Pseudocode for image based *K*-means clustering.

the number of moves between two states on the chessboard, as long as the states are not very far from each other in the search space. But there is no reason to assume that his judgments of similarity of two states that are two dozen (or more) moves away are expressed in the number of moves. The same is true when solving math or physics problems. Similarities among different areas of math and physics are expressed using a different metric than similarities within a given area. One surprising implication of this fact is that our multiscale model is able to provide a near-optimal solution to the TSP tour-length minimization problem, without ever measuring the length of the entire tour and without storing the entire tour in memory.

The rest of the process of producing successive TSP tour approximations is a recursive application of the process just described (see Figure 11 for the pseudocode of the recursion algorithm). The cluster in the bottom of the image contains 10 points which are represented in a 6×6 grid as three sub-clusters. When these three sub-clusters are inserted into the next approximation of the tour, the sub-cluster marked in blue is between the red and green because this satisfies the cheapest insertion criterion. Before the model analyzes the cluster on the top left, it has to increase the resolution of the sub-clusters in the bottom part of the image. First, it identifies three sub-sub-clusters in the red sub-cluster, representing the individual points, and decides about the order of their insertion. At this stage, the original TSP problem and the tour approximation is represented at three different levels of scale and resolution. Each level has its own distance metric and there is no need for

- Take photo of region using 6×6 pixel resolution camera. The output is a 6×6 grid of pixel intensity values between 0 and 1.
- If the sum of all of the intensities is below the intensity threshold
  - Return
- Use pixel intensity weighted *K*-means clustering to determine the child clusters.
- Remove the current point and *c* points before and after it.
- Use cheapest insertion to insert the following points into the tour:
  - The last *c* points before the current point (that were just removed).
  - The last *c* points after the current point (that were just removed).
  - The centroids of the child clusters.
- For each cluster in the order in which they appear in the tour
  - Recurse on the cluster. The cluster is the new region and the centroid is the new current point.
  - Write out any points which are not stored by higher levels of the recursion and are more than *c* points behind in the tour relative to the current cluster's centroid.
- Return.

Invariants maintained:

- The initial *c* points are always stored so that they can be used for cheapest insertion when the tour wraps around and reaches the end.
- Each level of the recursion only stores the cluster boundaries and centroids of the children
- The *c* points before and after the current point are always stored so that they are always available for the cheapest insertion.
- The points stored in the higher levels of the recursion might or might not intersect with the *c* points before the current point.
- The *c* points after the current point are always stored in the higher levels of the recursion or are part of the initial *c* points of the tour.

Figure 11.
Pseudocode of the recursion part of the new model.

these metrics to be identical. For example, the distances on a given level can be expressed using the size of one element of the 6×6 grid. Next, the same process is applied to the blue sub-cluster and then to the green one. Finally, the model is ready to analyze the cluster on top-left. After representing it on a 6×6 array, it puts two points in a single sub-cluster and the remaining four points become their own sub-clusters. After one more step, the TSP tour has been found. This tour happens to be the optimal TSP tour. It should be obvious that during the solution process our model produces only one valid TSP tour. It does not compare multiple tours in order to choose the best one. Comparing multiple TSP tours would be equivalent to a global search, which would always be very time consuming due to the combinatorial explosion. Global search is avoided through the use of a coarse to fine (global to local) series of tour approximations. The remaining search is performed locally when the cheapest insertion criterion is used. Figure 12 illustrates the main steps of the recursion algorithm.

Demo 2 (http://www.youtube.com/watch?v=Rkp7SyoE2r0) illustrates the same process on a problem with 69 cities. Again,

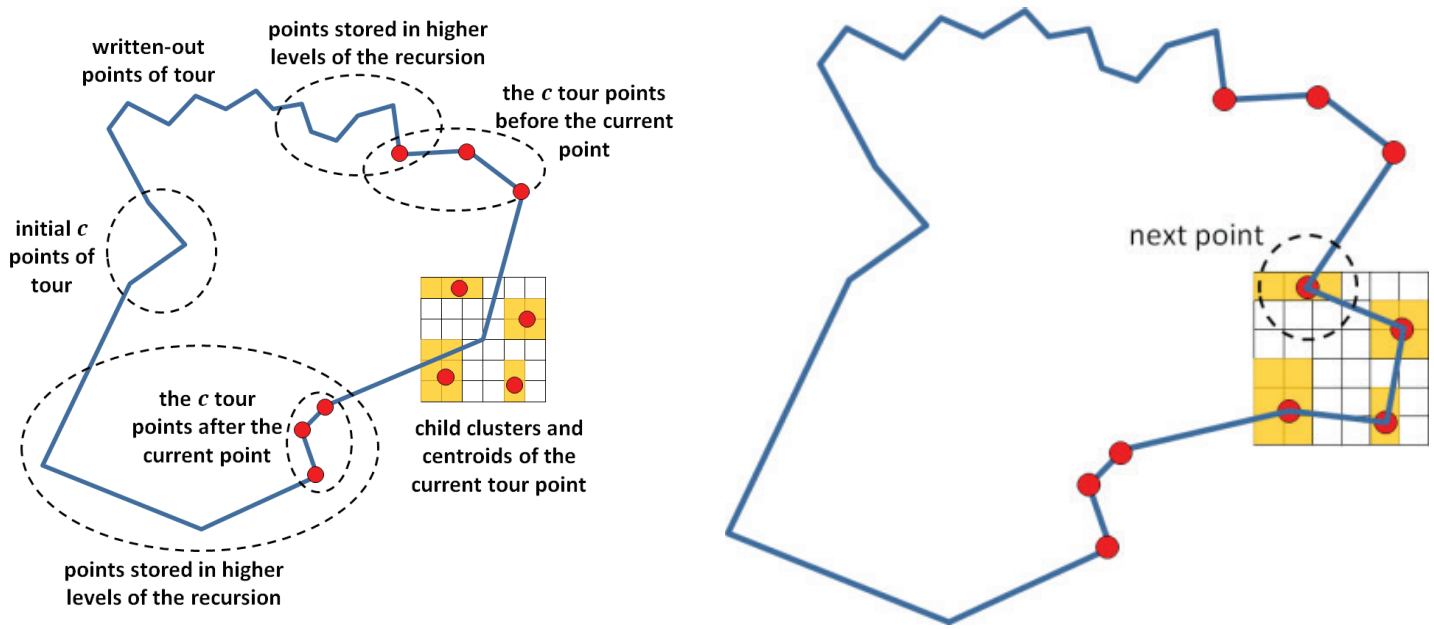

Figure 12.
Illustration of the algorithm before (left) a recursive step and after (right).

this problem was prepared by hand with a number of obvious clusters. The solution process is the same, except that the model had to use four levels of resolution. Demo 3 (http://www.youtube.com/watch?v=cvUghpLFV4I) illustrates the solution process with more than 100 points. Here, the model used five levels of resolution. The last demo (http://www.youtube.com/watch?v=h7ixhR9LpbM) shows a TSP problem with more than 100 points generated randomly with no obvious clusters. Again, the solution process is the same and the model used 5 levels of resolution. These TSP tours are all within 10–20% of error compared to the shortest tour. At the same time, the total number of memory units needed to store the current approximation of the TSP tour at any given time was not larger than 300. This performance level and memory usage stayed roughly the same when the number of cities was increased to 1,000: the maximal error was never greater than 25% and the memory required was never larger than 300 units (see Figure 13). In fact, the algorithm requires substantially less memory than 300 units. A TSP tour can be produced by keeping in memory only the 6×6 array representing the current cluster because all other values in memory can be recomputed on demand, assuming that our model can look at the TSP problem shown on a piece of paper in front of it as many times it needs using a camera, and that the part of the tour already produced has been drawn on this paper. This is how human subjects solve TSP problems.[1] It is known that *iconic* or sensory memory can store, for a duration equivalent to a fraction of a second an array of a dozen alphanumeric characters (Sperling, 1960). The visual system can store this amount of information for a brief amount of time and then select the relevant information for further processing. The selected information can be stored for a longer amount of time in working memory. The capacity of human working memory is smaller than the capacity of the visual memory and estimated as somewhere between four and nine elements (Miller, 1956; Baddeley, 1986). This corresponds quite well to the number of clusters in our algorithm. Indeed, the model solves the problem by using information about 2–5 clusters, rather than about the entire 6×6 array. We can conclude, therefore, that our model can solve large instances of TSP problem by storing only several pieces of information in its working memory, a requirement that matches quite well known properties of human working memory (Neath & Surprenant, 2003; Nairne & Neath, 2013). All other information about the problem can be acquired and computed on demand because the points representing the problem are in front of the model's camera, like they are in front of a human subject's eyes. The model can look at the next part of the problem when needed, and can remove the part of the tour that has already been solved.

We would like to point out that the key to the success of this model is its pyramid architecture. The pyramid architecture of this model agrees well with the known architecture of the human visual system (see the previous section). Furthermore, the computational complexity of this model is at most $n \log(n)$, where n is the number of cities. If at least some computations are done in parallel, as they surely are in the visual system, the complexity can be linear (Haxhimusa et al., 2011). This is similar to the known results with human subjects (see the section on human performance).

CONCLUSION

This is the first theory that can explain how a human mind can handle combinatorial optimizations problems with very large search spaces. The main requirement is that a

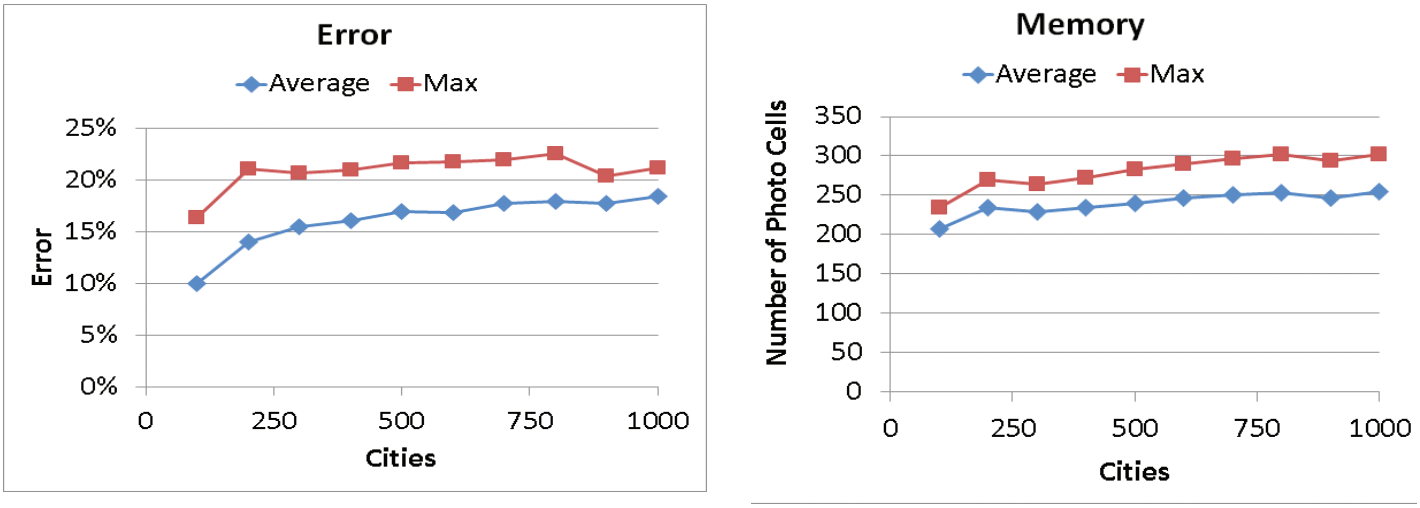

Figure 13.
Error and memory requirements of the version of the model, in which the entire current approximation to the TSP tour is stored. Average and maximal error are shown on left, and average and maximal memory usage as a function of the number of cities are shown on right. Each data point is based on 40 randomly generated TSP problems. Note that the model does not have to store the entire approximation of the TSP tour, but only the positions of 2–5 clusters that are being inserted into the already existing tour.

given problem can be represented at multiple levels of scale and resolution. Specifically, it must be possible to perform hierarchical clustering. But note that the clustering must be done in a top-down, coarse-to-fine direction. This is the only way to keep the memory requirements small. This, in turn, implies that there must be a way to judge similarities among states of the problem, as well as among clusters of the states (and clusters of clusters). Similarities are easy to formalize in TSP—this calls for nothing more than judging distances on the Euclidean plane. In more abstract problems, such as math and physics problems, a similarity metric will use characteristics of the underlying concepts. It seems that this is exactly how mathematics and physics are organized in our memory. This is also the way chess is played. An expert chess-player can judge whether or not two different situations on the chess board come from the same game, and also how far any two situations are without trying to build a complete representation of the problem space of chess consisting of an astronomical number of 10^{120} states. Using similarities (distances) among clusters and the relations between clusters and sub-clusters, a human can orient himself in large search spaces without being lost and without having to search the spaces. Hierarchical clustering provides us with a way to determine a direction in a problem space, which is essential in deciding where to go next without examining alternatives (Maier, 1930; Pizlo & Li, 2005; Chu, Li, Su, & Pizlo, 2010). As a result, we can quickly solve difficult problems nearly optimally, using working memory that stores only a few items at a time.

We conclude this paper with a few comments about how our model relates to existing psychophysical results, as well as to other models of TSP. Human subjects are typically tested with TSP problems containing a few dozen cities. There is at least one study in which subjects were tested with the number of cities as large as 120 (Dry et al., 2006). We can compare performance of our new model on 100 cities with that of subjects tested by Dry et al. The average error produced by our model when tested on 40 randomly generated TSP problems with 100 cities was 10% (see Figure 13) and the maximum error was 17%. This is quite similar to the performance of the subjects. Dry et al.'s Figure 2 shows that the average error computed from 40 subjects, when each subject solved a 100 city TSP once, was about 11% and standard error was about 1%. With 40 measurements per data point, this implies that standard deviation of the error was about 6%. The maximum error produced by our model is about the same as the average error produced by subjects plus one standard deviation. In the absence of a more direct test, this comparison suggests that human subjects might be using an algorithm like that described here.

Could our idea of using a small size of working memory be implemented with other models of how humans solve TSP, for example with MacGregor & Ormerod's (1996) model that starts with finding a convex hull of the points and then proceeds with a cheapest insertion? It is possible, but it seems that in order to do this, their model (or any other model of TSP) would have to at some point include a hierarchical clustering aspect of our model. Otherwise, analyzing only a small fraction of a problem at a time would make it greedy, the same way a Nearest Neighbor (NN) algorithm is greedy. We already know that human subjects systematically outperform NN algorithm. So, it seems to us, that the only way to produce small errors by using a small sized working memory is to use a pyramid architecture.

One of the reviewers pointed out that considering the growing amount of empirical evidence about the role of convex hull, self intersections of a tour, relative clustering/regularity of points, as well as considering the existence of other models of how humans solve TSP, one should perform a direct model-to-model comparison against empirical data. We completely agree and we will perform such tests in near future.

NOTES

1. Instead of presenting TSP on a piece of paper, it can be presented on a computer monitor.

REFERENCES

Baddeley, A. (1986) *Working memory*. Oxford: Clarendon Press.

Brusco, M. J., & Stahl, S. (2005). *Branch-and-bound applications in combinatorial data analysis*. New York: Springer.

Chu, Y., Li, Z., Su, Y., & Pizlo, Z. (2010). Heuristics in problem solving: the role of direction in controlling the search space. *Journal of Problem Solving, 3*(1), 27–51. http://dx.doi.org/10.7771/1932-6246.1078

Dry, M., Lee, M. D., Vickers, D., & Hughes, P. (2006). Human performance on visually presented Traveling Salesperson Problems with varying numbers of nodes. *Journal of Problem Solving, 1*(1), 20–32. http://dx.doi.org/10.7771/1932-6246.1004

Duncker, K. (1945). On problem solving. *Psychological Monographs*, *58*.

Gardner, H. E. (1987). The mind's new science: A history of the cognitive revolution. New York: Basic Books.

Graham, S. M., Joshi, A. & Pizlo, Z. (2000). The Traveling Salesman Problem: a hierarchical model. *Memory & Cognition*, *28*, 1191–1204.

Haxhimusa, Y., Carpenter, E., Catrambone, J., Foldes, D., Stefanov, E., Arns, L., & Pizlo, Z. (2011). 2D and 3D Traveling Salesman Problem. *Journal of Problem Solving, 3*(2), 167–193. http://dx.doi.org/10.7771/1932-6246.1096

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1985). *The Traveling Salesman Problem*. New York: Wiley.

Lihoreau, M., Chittka, L. & Raine, N. E. (2010). Travel optimization by foraging bumblebees through readjustments of traplines after discovery of new feeding locations. *The American Naturalist*, *176*, 744–767. http://dx.doi.org/10.1086/657042

MacGregor, J., & Ormerod, T. (1996). Human performance on the traveling salesman problem.
Perception & Psychophysics, 58, 527–539. http://dx.doi.org/10.3758/BF03213088

Maier, N. R. F. (1930). Reasoning in humans: I. On direction. *Journal of Comparative Psychology, 10*, 115–143. http://dx.doi.org/10.1037/h0073232

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, *63*, 81–97. http://dx.doi.org/10.1037/h0043158

Nairne, J. S., & Neath, I. (2012). Sensory and working memory. In A. F. Healy & R. W. Proctor (Eds.), *Comprehensive handbook of psychology* (2nd ed.), *Vol. 4: Experimental Psychology* (pp. 419–445). New York: Wiley.

Neath, I., & Surprenant, A. M. (2003). *Human memory: An introduction to research, data, and theory* (2nd ed.). Belmont, CA: Wadsworth.

Pizlo, Z., & Li, Z. (2005) Solving combinatorial problems: 15-puzzle. *Memory & Cognition, 33,* 1069–1084. http://dx.doi.org/10.3758/BF03193214

Pizlo, Z., Rosenfeld, A., & Epelboim, J. (1995). An exponential pyramid model of the time course of size processing. *Vision Research*, *35*, 1089–1107. http://dx.doi.org/10.1016/0042-6989(94)00195-R

Pizlo, Z., Stefanov, E., Saalweachter, J., Li, Z., Haxhimusa, Y., & Kropatsch, W. G. (2006). Traveling Salesman Problem: A foveating pyramid model. *Journal of Problem Solving*, *1*(1), 83–101. http://dx.doi.org/10.7771/1932-6246.1009

Sperling, G. (1960). The information available in brief visual presentations. *Psychological Monographs: General and Applied*, *74*, 1–29. http://dx.doi.org/10.1037/h0093759

Wandell, B. A. (1995). *Foundations of vision*. Sunderland, MA: Sinauer.

Wertheimer, M. (1945). *Productive thinking*. New York: Harper & Brothers.